Applied Mathematics and Computation 267 (2015) 304–313

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

On the use of multiple heterogeneous devices to speedup the execution of a computational model of the Human Immune System

T.M. do Nascimento, J.M. de Oliveira, M.P. Xavier, A.B. Pigozzo, R.W. dos Santos, M. Lobosco *

FISIOCOMP – Laboratory of Computational Physiology and High Performance Computing, Federal University of Juiz de Fora, Brazil

ARTICLE INFO

Keywords:
HPC
APU
PDE
Computational immunology

ABSTRACT

The Human Immune System (HIS) is responsible for protecting the body against diseases, but the mechanisms used in this task are not completely understood. Mathematical and computational tools can be used for this purpose, and due to the costs involved in simulating the HIS, GPUs (Graphics Processing Units) are frequently used as the computational platform. The frequency in which GPUs are used for tasks like this is so high that some processing units, such as the APUs (Accelerated Processing Units), have integrated then into the CPU chip. This work presents the implementation on an APU of a mathematical model that describes part of the HIS. A load balancing strategy was implemented to distribute data with the objective of equalizing the load at each computational device, since GPU and CPU are heterogeneous. Gains up to 6.0×, 1.28× and 3.7× were obtained by the balanced version of the code, when compared to the same parallel versions that execute exclusively on CPU, GPU, and on both of them, but without using load balancing, respectively.

1. Introduction

Parallelism is now widely available: CPUs are composed of multiple cores, GPUs are composed by hundreds or even thousands of them, and APUs (Accelerated Processing Units) [1] are available using both technologies in a single chip. Despite the availability of such huge number of cores, sometimes they are not entirely explored by parallel programs due to their heterogeneous nature, which makes hard to develop codes that use them simultaneously.

In this scenario, OpenCL [2] offers a framework to develop parallel programs that allows simultaneous use of multiple heterogeneous devices to accelerate the execution of programs. However, it is the programmer's responsibility to implement his/her own load balancing strategy.

This work presents results of the execution on an APU of a parallel computational implementation of a mathematical model [3] that describes part of the Human Immune System (HIS). A set of PDEs is used to describe the dynamics of different cells and molecules on a three dimensional section of tissue. This tissue is discretized into multiple points, and at each time step the same set of equations are computed for each point, using for this purpose distinct data.

* Corresponding author.
E-mail addresses: tiago.nascimento@uab.ufjf.br (T.M. do Nascimento), jmarcosdo@gmail.com (J.M. de Oliveira), micaelpx@gmail.com (M.P. Xavier), alexbprr@gmail.com (A.B. Pigozzo), rwdsantos@yahoo.com (R.W. dos Santos), marcelo.lobosco@ice.ufjf.br (M. Lobosco).

http://dx.doi.org/10.1016/j.amc.2015.03.072
0096-3003/

The number of points and time steps used in the simulation, as well as the type and amount of computation that must be performed for each point determine the total computation time. Results of particular interest for biologist demands a large amount of points or time steps with huge computational requirements, justifying the use of accelerators such as those available on APUs. OpenCL has been used to implement the parallel version of the code that executed simultaneously on all CPU and GPU cores available for use: 4 CPUs cores and 384 GPUs cores. In order to deal with the heterogeneity of the APU architecture, a load balancing strategy was implemented. The use of all devices available accelerate the code up to 6 times when compared to the parallel version that executes using only the CPU cores and about 28% when compared to the parallel version that uses only the GPU cores. Also, the load balancing strategy implemented was able to accelerate the code about 3.7 times when compared to the unbalanced version of the code. It must be stressed that the technique presented in this work is general, in the sense that it can be used also to improve the performance of other applications.

The remaining of this work is organized as follows. Section 2 presents an overview of the HIS and the set of PDEs used in this work to describe it. Section 3 presents OpenCL in short. Section 4 presents the implementation details and Section 5 presents the results obtained. Finally, the last Section presents the conclusions and plans for future works.

2. Innate Human Immune System

2.1. Biological background

The HIS is responsible for many distinct tasks. Probably its most known task is to act as a defense mechanism: it is responsible for protective response against lots of potentially pathogenic microorganisms [4]. This defense against pathogenic microorganisms has been divided into two general types of responses: innate and adaptive immune response. The innate immune response is started when a microorganisms that breaks our first line of defense, the skin or the mucous membrane, interacts with some types of cells, macrophages, that are resting in the tissue and recognize some key molecules present on many microorganisms, such as lipopolysaccharides (LPS). As a result, macrophages became active or, in other words, it enhances its paghocytic capability. Also, some molecules, called cytokines, are produced to signal that an invasion has begun. This pro-inflammatory substance is responsible for recruiting more cells, such as neutrophils, to the site where invaders are located. These cells migrate basically from blood vessels. Neutrophils are short-lived cells that also phagocytose and kill pathogens using several mechanisms[4]; they secrete protein granules that attract macrophages to the site of infection and also increase the endothelium permeability. Therefore more immune cells can leave the vasculature and enter into the tissue. Neutrophils die due to the production of reactive oxygen species (ROS) or through constitutive apoptosis.

Dendritic cells are also recruited, and its role is to uptake parts of the microorganisms that was phagocyted and transport them to the lymph nodes, where they can activate the adaptive immune system. The key characteristic of the adaptive immune system is the ability to adapt its defense cells to protect against a specific invader. It is composed mainly of B and T cells. B cells produce antibodies, that tag invaders for destruction (a process called opsonization) by phagocyte cells such as the macrophages. However, the defense mechanism described so far has a flaw: if a microorganism, such as a virus, gets into a cell, the antibodies cannot opsonise it, nor the phagocytes can destruct it. If a virus gets into a cell, it can make thousands of copies of itself[5]. T cells plays an important role in this case since they can recognize and kill infected cells.

After the HIS has eliminated all pathogenic microorganisms, it has to finish the inflammatory process. The resolution of the inflammatory response is a complex process that includes the production of anti-inflammatory mediators and the apoptosis of effector cells of the HIS, such as neutrophils.

2.2. Mathematical model

The complete modeling of the HIS demands a huge amount of work. In this work, the APU is used to solve a system of PDEs that models the innate HIS. The main motivation to model the innate HIS is to propose answers to open questions as well as to propose new important questions. For example, chronic bacterial infections are characterized by being a large bacterial infection and/or an infection where the bacteria grows rapidly. In these cases, the immune response is not capable of completely eliminating the infection which may lead to the formation of a pattern known as abscess, an area comprising fluids, bacteria, immune cells (mainly neutrophils), and dead cells [6]. The factors that lead to the abscess formation are not completely understood. Mathematical and computational tools could help to improve the understanding of such process.

The mathematical model depicted in Fig. 1 simulates the temporal and spatial behavior of lipopolysaccharide (LPS), that represents an antigen in our model, neutrophils (N), apoptotic neutrophils (ND), pro-inflammatory cytokines (CH), anti-inflammatory cytokine (AC) and protein granules (G). Macrophages are present in two states of readiness: resting (RM) and hyperactivated (AM). A set of eight PDEs was used to implement this model [3] and are reproduced here to better contextualize the paper. A detailed discussion about them can be found in [7].